

REMARKS

The above Amendments and these Remarks are in reply to the Office Action mailed July 15, 2008.

Currently, claims 1-3, 5-15, 17-24, 26-30, 32-35, 37-42 and 44-49, 51-54 are pending. Applicants have amended claims 2, 3, 5-12, 14-15, 17-21, 23-24, 26-30, 32, 34-35, 37-38, 41-42, 44-49, and 51-52. Claims 31 and 50 have been cancelled. Claims 53-54 have been added. No new matter has been added as a result of these claim amendments. Applicants respectfully request reconsideration of claims 1-3, 5-15, 17-24, 26-30, 32-35, 37-42 and 44-49 and consideration of new claims 53-54.

Claim Objections

Claims 2, 3, 5-12, 12-15, 17-21, 23-34, 26-32, 34, 35, 37-38, 41-42, and 44-52 are objected to because of informalities.

The Applicants have amended the preamble of claims 2, 3, 5-12, 12-15, 17-21, 23-29, 31-34, 26-32, 34, 35, 37-38, 41-42, and 44-49, 51-52 to correct these informalities. Specifically, the Applicants have changed "A process" to "The process" in dependent claims 2, 3, 5-12, 12-15, 17-21 to indicate that the process in these dependent claims refers back to the process in claim 1. Similar amendments have been made to other dependent claims.

Claim 49 was objected to because the response filed on September 9, 2007 failed to mark the changes made to claim 49. At page 2 of the Office Action, the Examiner indicated that the amendment to claim 49 in the response filed on September 9, 2007 was entered and that claim 49 is being considered, but is objected to. Markings to claim 49 have been made to show the amendments to claim 49 in the response filed on September 9, 2007. Also note that the preamble of claim 49 was amended in this response to address the objection to the preamble. The markings to claim 49 are relative to the version of the claim that was submitted in the response filed on April 23, 2007.

Rejection of Claims 1-3, 5-8, 10, 12-15, 17-18, 20, 39-42, 44, and 47-52 Under 35 U.S.C. §103(a)

Claims 1-3, 5-8, 10, 12-15, 17-18, 20, 39-42, 44, and 47-52 have been rejected under 35 U.S.C. §103(a) as being unpatentable over Berkley (US 6,351,843) in view of Webster (US 6,738,965). The rejection is respectfully traversed for the following reasons.

Claim 1 recites:

A process for monitoring, comprising:
accessing a method;
determining whether to modify said method, said step of determining whether to modify said method includes determining whether said method calls another method; and
modifying said method for a particular purpose if said method calls another method.

Neither Berkley nor Webster, alone or in combination, disclose “said step of determining whether to modify said method includes determining whether said method calls another method,” as claimed.

Berkley fails to disclose “said step of determining whether to modify said method includes determining whether said method calls another method,” as claimed. The Office Action on page 8 appears to concede that Berkley fails to disclose determining whether a method calls another method. However, the Office Action also asserts the Berkley discloses a “call stack” which **can be used** to determine a method that calls other methods (page 4, line 12-15). However, the mere fact that a call stack could be used to determine which methods call other methods in no way teaches **determining which methods to modify** based on which methods call another method. Berkley discloses a technique for inserting a function into an existing application executable (col. 2, lines 18-19), which the Office Action appears to interpret as modifying a method. However, Berkley determines where to insert the function by checking configuration settings (col. 2, lines 17–55) not by “determining whether said method calls another method.” Therefore, Berkley does not teach these claim limitations.

The Office Action appears to allege that modifying Webster overcomes this deficiency in Berkley. First Applicants assert that Webster fails to disclose “said step of determining whether to modify said method includes determining whether said method calls another method,” as claimed.

Webster discloses that a user can specify “methods of interest” to trace. Webster discloses that the user can specify those methods in a file (col. 5, lines 52-57). Webster discloses that the methods can be specified either by class or method name (col. 6, lines 5-7). However, merely stating that a method that a user desires to trace can be specified by method name does not equate to disclosing determining to modify a method based on whether the method calls another method. In other words, Webster does not disclose anything about determining which methods to modify much less modifying a method based on whether the method calls another method.

The Office Action appears to reason that Webster discloses that a user can specify which methods to trace according to some purpose or interest of the user. Further, the Office Action appears to assert that it is well known in the computer arts that one such purpose or interest of the user is methods calling other methods (page 4, lines 6-9 and page 9, lines 2-4). Applicants assert that it is not sufficient to allege that it is well known in the art that a method may call another method and therefore it **is of interest to modify the calling method**. That is, just because someone of ordinary skill in the art is aware that one method calls another method does not mean that it is well known in the art that it would be of interest to modify the calling method. In other words, there is a large gap between knowing that a method calls another method and deciding to modify the calling method because it calls another method. Applicants respectfully assert that it is improper to attempt to fill in this gap by taking “Official Notice.” Applicants respectfully assert that **the Office Action presents no documentary evidence** to support that it is well known in the computer arts to determine whether to modify a method based on whether or not a method calls another method.

Thus, the Applicants respectfully assert that the Office Action is improperly relying upon Official Notice. As stated in MPEP 2144.03 “[o]fficial notice unsupported by documentary evidence should only be taken by the examiner where the facts asserted to be well-known, or to be common knowledge in the art are capable of **instant and unquestionable demonstration** as being well-known.” As noted by the court in *In re Ahlert*, 424 F.2d 1088, 1091, 165 USPQ 418, 420 (CCPA 1970), the notice of facts beyond the record which may be taken by the examiner must be “capable of such instant and unquestionable demonstration as to defy dispute” (citing *In re Knapp Monarch Co.*, 296 F.2d 230, 132 USPQ 6 (CCPA 1961)). Applicants respectfully assert that it is not instantly and unquestionably known to modify a method based on whether or not the method calls

another method. Furthermore, as noted by the court in *Ahlert*, any facts so noticed should be of notorious character and serve only to "fill in the gaps" in an insubstantial manner which might exist in the evidentiary showing made by the examiner to support a particular ground for rejection. It is never appropriate to rely solely on common knowledge in the art without evidentiary support in the record as the principal evidence upon which a rejection was based. See *Zurko*, 258 F.3d at 1386, 59 USPQ2d at 1697; *Ahlert*, 424 F.2d at 1092, 165 USPQ 421. Applicants respectfully assert that the Office Action is not merely "filling in the gaps" in an **insubstantial** manner. To the contrary, the documentary evidence that is missing is **substantial**.

Therefore, the Applicants demand that the Examiner either produce documentary evidence that it is well known in the art to **determine which methods to modify** based on which methods call another method or drop the rejection and allow claim 1.

Furthermore, Berkley discloses monitoring classes in an object-oriented environment and **cannot be modified** to trace on a method by method basis. Berkley discloses monitoring classes in an object-oriented environment. A class may include one or more objects, and the class defines how an object is implemented. Objects are instances of classes (col. 4, lines 61-62). Each object may include one or more methods, as shown in Figure 1. Therefore, a class may contain one or more objects, which each may contain one or more methods. As shown in Figure 1, a method may or may not call another method. For example, in Object 1, which is part of a class, Method A calls Method B. However, Method D does not call any other method.

Berkley discloses monitoring methods by class as follows. If a user wishes to monitor a particular method within a program, the user can change configuration settings associated with the program to specify or activate the class of the method that the user wishes to monitor. "[C]onfiguration settings 300 received as part of a program's execution code are modified at runtime to add a setting to specify (as one example) tracing for a desired class of the executable 310, thereby producing new configuration settings 320" (Berkley Figure 5 and col. 6, lines 52-53). Berkley further explains that:

the new configuration settings 330 are then employed with the existing application executable 340 to run the application executable 350. The object runtime will query the configuration settings and when a trace is active for a given class, will dynamically insert the trace class into the inheritance hierarchy for that class... Runtime will then see the trace class within the hierarchy and setup for a trace 360 (Berkley col. 6, line 64 – col. 7, line 4).

For example, for a class named 'class 1' that is specified in the configuration settings, the trace is set up through "redirection stubs [that] are created [to] trace entry and exit methods around each target method within class 1" (Berkley col. 7, lines 61-63). A trace is active for a given class when the user specifies the class in the configuration settings. The example for activating classes in column 7, lines 20-26 of Berkley shows:

configuration settings used to set up a desired tracing environment. A configuration variable is initialized with the name(s) of the class(es) whose methods should be traced:

[TraceOptions]

ClassTrace = Class1, Class2, Class3, Class4

In the above example, Class1, Class2, Class3, and Class4 are set active by specifying these classes in the ClassTrace list. **All of the methods** in these classes will be traced. Using Figure 1 of Berkley as an example, if Object 1 was contained within Class 1, which is specified in the ClassTrace list above, all of the methods contained within Object 1 would be modified for tracing, even Method D which does not call any other method. Since all of the methods are traced, Berkley cannot be modified to arrive at the claimed invention.

As the foregoing clearly states, Berkley discloses that tracing performed by activating **classes**. Berkley does not disclose a technique for a user to specify a particular method to trace. Therefore, Berkley cannot be modified to trace individual methods.

For all of the foregoing reasons, claim 1 is patentable over the cited prior art. Independent Claims 13, and 39 each contain a similar feature and are therefore patentable over the cited prior art for at least the same reasons as claim 1. Therefore, these claims are believed to be allowable.

Currently amended claim 47 recites:

A process for monitoring, comprising:
 accessing a method;
 determining whether said method is complex, said step of determining includes determining that said method is complex if said method satisfies at least one of the following criteria:
 said method calls another method;
 said method has an access level that is either public or package; or
 said method is not flagged by a compiler as being synthetic; and
 adding a tracer to said method only if said method is determined to be complex.

For reasons discussed in the response to claim 1, neither Berkley nor Webster, alone or in combination, disclose “said step of determining includes determining that said method is complex if said method satisfies at least one of the following criteria ... said method calls another method... adding a tracer to said method only if said method is determined to be complex,” as claimed.

Neither Berkley nor Webster, alone or in combination, disclose “said step of determining includes determining that said method is complex if said method satisfies at least one of the following criteria ... said method has an access level that is either public or package ... adding a tracer to said method only if said method is determined to be complex,” as claimed. The Office Action asserts that it is well known that the Java specification defines flags that indicate whether the access level of a method. Even if this is true, there is no teaching in the prior art **to determine that a method is complex if said method has an access level that is either public or package and to add a tracer to said method only if said method is determined to be complex**. Thus, there is more missing from the teaching of the prior art than the Office Action concedes on page 9. Additionally, for reasons discussed in the response to claim 1, Berkley cannot be modified to add a tracer to said method only if said method is determined to be complex for the case in which the criteria for a method being complex is if said method has an access level that is either public or package.

Neither Berkley nor Webster, alone or in combination, disclose “said step of determining includes determining that said method is complex if said method satisfies at least one of the following criteria ... said method is not flagged by a compiler as being synthetic ... adding a tracer

to said method only if said method is determined to be complex,” as claimed. The Office Action asserts that it is well known that synthetic methods generated by a Java compiler are flagged and thus easily identifiable. Even if this is true, there is no teaching in the prior art **to determine that a method is complex if it is not flagged by a compiler as being synthetic and to add a tracer to said method only if said method is determined to be complex**. Thus, there is more missing from the teaching of the prior art than the Office Action concedes on page 9. Additionally, for reasons discussed in the response to claim 1, Berkley cannot be modified to add a tracer to said method only if said method is determined to be complex for the case in which the criteria for a method being complex is that it is not flagged as being synthetic.

Webster does not remedy the foregoing deficiencies in Berkley.

For the foregoing reasons, claim 47 is patentable.

Rejection of Claims 9, 11, 19, 21-24, 26-35, 37, 38, 45, and 46 Under 35 U.S.C. §103(a)

Claims 9, 11, 19, 21-24, 26-35, 37, 38, 45, and 46 have been rejected under 35 U.S.C. §103(a) as being obvious over Berkley in view of Webster in further view of Berry (US 6,662,359). Because the cited prior art, alone or in combination, does not teach or suggest all of the limitations of the rejected claims, Applicants assert that the claims are in condition for allowance.

Berkley in view of Webster, as discussed above, does not disclose “determining whether to modify said method, said step of determining whether to modify said method includes determining whether said method calls another method,” as recited in claim 1. Claims 9, 11, 19, 21-24, 26-35, 37, 38, 45, and 46 all either contain a similar feature or depend from an independent claim that recites a similar feature.

Additionally, Berry does not teach or suggest this feature. Instead, Berry discloses inserting a hook into a class when an exception is called so that the method throwing the exception can be identified, yet no determination based on “whether [a] method calls another method” is disclosed. Therefore, the combination of Berkley and Berry does not disclose, teach, or suggest all of the limitations of claims 9, 11, 19, 21-24, 26-35, 37, 38, 45, and 46. Applicants respectfully request reconsideration of these claims.

New Claims

Claims 53-54 have been added. Claim 53 further limits claim 47 by recited that the determining that said method is complex if said method satisfies at least two of said criteria. Applicant asserts that the prior art does not teach these limitations. Claim 54 further limits claim 47 by recited that the determining that said method is complex if said method calls another method. For the reasons discussed in the response to claim 1, Applicant asserts that the prior art does not teach these limitations.

Based on the above amendments and these remarks, reconsideration of claims 1-3, 5-15, 17-24, 26-29, 31-35, 37-42 and 44-49, 51-52 and consideration of new claims 53-54 is respectfully requested.

The Examiner's prompt attention to this matter is greatly appreciated. Should further questions remain, the Examiner is invited to contact the undersigned attorney by telephone.

The Commissioner is authorized to charge any underpayment or credit any overpayment to Deposit Account No. 501826 for any matter in connection with this response, including any fee for extension of time, which may be required.

Respectfully submitted,

Date: November 11, 2008

By: /RonaldMPomerence/
Ronald M. Pomerence
Reg. No. 43,009

VIERRA MAGEN MARCUS & DENIRO LLP
575 Market Street, Suite 2500
San Francisco, California 94105-4206
Telephone: (415) 369-9660
Facsimile: (415) 369-9665